Introduzione alle Reti Neurali

Gianluca Baldassarre

Introduzione e cenni storici

Le reti neurali sono un modello quantitativo prodotto dall'intelligenza artificiale, ispirato alla struttura ed al funzionamento del cervello. Le reti neurali sono un modello di interesse per la psicologia in quanto possono essere utilizzate per rappresentare gli scambi di segali tra neuroni, le attivazioni dei neuroni e la plasticità delle sinapsi. Introduciamo le reti neurali attraverso accennando alla storia del loro sviluppo.

L'idea di rete neurale formata da unità collegate tra loro da connessioni similmente a come i neuroni sono collegati tra loro dalle sinapsi, viene proposta per la prima volta da *McCulloc e Pitts (1943)* con l'eplicito intento di mimare il funzionamento del cervello. Le loro reti sono formate da unità binaria a soglia, come quelle che vedremo tra poco, in grado di eseguire semplici operazioni logiche (la rete nel suo complesso si comporta come una rete ad unità booleane che vedremo tra poco). Si tratta del primo modello matematico del cervello ad essere direttamente ispirato dalla struttura del cervello stesso. Questo modello è importante per diverse ragioni. Innanzi tutto esso mostra che è possibile costruire un modello formale matematico del cervello, e questo implica l'importante assunzione che i processi psicologici possono essere studiati a partire dalle operazioni quantitative dei neuroni del cervello. In secondo luogo il modello sviluppa i concetti di elaborazione parallela e di rappresentazione distribuita che vedremo in seguito. Uno dei risultati più importanti mostrati con questo modello è che le reti booleane formate da unità molto semplici, possono svolgere qualsiasi tipo di processamento di informazione (sarebbe necessario fare riferimento ad a dei concetti di teoria della computazione, che esulano da questo scritto, per poter chiarire in modo rigoroso questi concetti).

Un secondo evento storico importante per lo sviluppo delle reti neurali è la proposta dell'idea per cui apprendimento delle reti neurali sia basato sulla modifica dei pesi delle connessioni tra unità (vedi più sotto). Questa idea nasce con l'importante ipotesi dello psicologo *Hebb (1948).* La regola di Hebb dice che se due unità si attivano contestualmente, la connessioni esistenti tra loro si rafforzano. Questo comporta che se in un secondo momento una delle due unità si attiva, essa invia un impulso forte all'altra unità che tende quindi anch'essa ad attivarsi. Questo meccanismo è importante in quanto formalizza in modo preciso l'idea di associazione (tra stimoli, atti motori, concetti, ecc.) elaborata precedentemente in filosofia e in psicologia su un piano puramente concettuale/verbale.

In seguito alla comprensione dettagliata del funzionamento del singolo neurone realizzata studiando sperimentalmente l'assone gigante del calamaro, *Hodgkin e Huxley (1952)*, due neurofisiologi, formulano un modello matematico rigoroso del singolo neurone. Questo modello è costituito da più equazioni che descrivono nel dettaglio l'andamento dei potenziali elettrici e l'insorgere dello spike nei neuroni. Questo modello è il punto di partenza dei modelli, detti a *spiking neurons*, attualmente utilizzati dalla *computational neuroscience*, una branca di ricerca interdisciplinare che collega le neuroscienze con la teorizzazione basata su modelli matematici precisi dei neuroni e delle mappe e nuclei neuronali.

Un passo importante nella storia delle reti neurali è la proposta da parte dello psicologo *Rosenblatt (1958)* del Perceptrone, una rete neurale in grado di apprendere dai propri errori sulla base di un apprendimento supervisionato. L'apprendimento supervisionato comporta che un insegnante

esterno alla rete fornisca ad essa il pattern di output corretto (o “desiderato”) che la rete dovrebbe emettere in corrispondenza di ogni particolare pattern di input. La rete, sulla base delle differenze tra l’output desiderato e l’output che essa ha emesso (queste differenze si chiamano errori), cambia i propri pesi sulla base di una regola chiamata Regola Delta, in modo da diminuire queste differenze stesse. Il perceptrone ha delle unità di uscita binarie (illustrate in seguito). La generalizzazione della regola delta ad unità di uscita sigmoidali (illustrate in seguito) è stata effettuata da *Widrow e Hoff (1960)* medianate la proposta della regola di Widrow-Hoff. Il perceptrone consente di modellare processi cognitivi di classificazione, riconoscimento, comportamenti senso-motori, processi di associazione e di memorizzazione.

Negli anni ’60 si assiste al boom dell’intelligenza artificiale simbolica basata sulla manipolazione di simboli qualitativi con regole della logica, anziché su computazioni quantitative come avviene nelle reti neurali. Questo porta rapidamente alla costruzione di programmi intelligenti in grado di partecipare a giochi da tavolo, di risolvere problemi, di pianificare sequenze di azioni, di fornire pareri ed indicazioni in campi precisi della conoscenza (sistemi esperti). Il successo di questi programmi crea entusiasmo anche all’interno della psicologia e contribuisce a diffondere l’uso di questi programmi come modelli dei processi cognitivi umani di alto livello. Questo porta quindi all’avvento dell’approccio information processing della psicologia cognitiva, dove il computer con la sua architettura ed il suo funzionamento diviene il modello-metafora di riferimento per lo studio delle funzioni cognitive. Inoltre il successo dei programmi di intelligenza artificiale porta alla creazione di una branca di ricerca interdisciplinare che coinvolge la psicologia, l’intelligenza artificiale, le teorie della mente della filosofia ed altre discipline, appellate nel loro insieme come *scienze cognitive*.

Il successo dell’intelligenza artificiale basata sulla logica, e l’avvento delle scienze cognitive, porta anche ad una battuta di arresto delle ricerche sulle reti neurali. Questa fase di arresto è anche segnata da uno specifico evento storico, il forte attacco alle potenzialità delle reti neurali, ed in particolare del perceptrone, condotto con uno studio teorico da parte di Minsky e Papert (1969). Questi autori mostrano che il perceptrone non è in grado di risolvere un importante classe di problemi di classificazione chiamati problemi non linearmente separabili. L’effetto di questo studio fu dirompente, probabilmente sulla scia del contesto storico segnato dai successi delle scienze cognitive. Nel giro di pochi anni il numero di ricercatori sulle reti neurali decresce drammaticamente ed ha inizio un periodo di medioevo per l’avanzamento della disciplina.

Pochi ricercatori continuano comunque lo studio delle reti neurali introducendo dei concetti che riveleranno la loro importanza solo più tardi. Tra questi Grossberg (1969 - 1980) effettua una serie di studi su reti neurali in grado di mediare tra le opposte esigenze della plasticità e rigidità, ovvero tra l’esigenza di apprendere nuovi comportamenti e nello stesso tempo non stravolgere quanto appreso in precedenza. Queste reti vengono utilizzate per modellizzare una serie di processi mentali quali i processi di apprendimento, di pensiero, di immaginazione e di controllo dei movimenti.

Un altro contributo importante è lo studio di *Kohonen (1981)* che porta a proporre la rete di Kohonen in grado di auto-organizzarsi senza un supervisore esterno, cioè di formare delle categorie dei pattern di input sulla base delle loro proprietà statistiche (ad esempio la loro somiglianza).

La rinascita dell’interesse diffuso nelle reti neurali si ha con gli studi di *Hopfield (1982)* che propone la rete di Hopfield una rete che sulla base della regola di Hebb è in grado di modellare in modo complesso la memoria associativa e di risolvere importanti problemi di ottimizzazione. Il funzionamento della rete di Hopfield viene anche studiato rigorosamente ed in modo approfondito sulla base degli strumenti matematici utilizzati dai fisici per lo studio dei *vetri di spin*. Un vetro di spin è un corpo metallico contenente delle impurità e costituito da degli atomi con un orientamento

magnetico casuale in condizioni normali. L'aspetto interessante dei vetri di spin è che gli atomi di cui essi sono composti si allineano in modo ordinato quando la temperatura va oltre un certo livello critico, sulla base di processi di auto-organizzazione spontanea, cioè senza un co-ordinamento centralizzato. Questi studi conferiscono alle indagini sulle reti neurali un notevole solidità teorica che portò al rilancio dell'interesse per esse e ad una produzione notevole di ricerche.

Il boom dell'interesse per le reti neurali si ha comunque con la pubblicazione di due volumi da parte di un gruppo di ricercatori statunitensi (chiamato Parallel Distributed Processing group, o PDP group) tra cui Rumelhart, Hinton, McClelland, e Williams (1986), che presentano una ampia gamma di ricerche atte mostrare le proprietà generali delle reti neurali, alcuni algoritmi specifici di apprendimento, e una serie di modelli dei processi psicologici. Uno degli algoritmi proposti, l'algoritmo di apprendimento Error-Back Propagation, ha consentito di superare i problemi messi in luce dal lavoro di Minsky e Papert e costituisce l'algoritmo più potente ed usato nelle applicazioni che utilizzano le reti neurali. Nel complesso questi contributi hanno consentito di affermare definitivamente e dare una chiara identità al nuovo paradigma di ricerca di intelligenza artificiale basato sulle reti neurali. Questo paradigma viene chiamato ora *connessionismo* (perché l'elemento base delle reti neurali sono le connessioni).

Analisi matematica in pillole: potenze e radici, funzioni matematiche, grafici di funzioni

Per poter comprendere cosa sia una rete neurale, è necessario sapere cosa sia una variabile ed una funzione matematica.

Una *variabile matematica* è un simbolo che indica un numero che assumere una certa gamma di valori, e viene di solito indicata con una o più lettere come: x, y, z, A, VMax.

Una *funzione matematica* è una relazione matematica che indica come variano alcune variabili, dette variabili dipendenti, al variare di altre variabili, dette variabili indipendenti. Ad esempio, una funzione matematica è questa:
y = 2 * x
(di solito in matematica il simbolo della moltiplicazione * viene omesso perché due simboli giustapposti si intendono moltiplicati tra loro. In tal caso la funzione diventa: y = 2 x). In questa espressione x ed y sono delle variabili matematiche che possono assumere vari valori numerici, ad esempio 5 -7 2,5 –16234,42342. La nostra funzione matematica ci dice appunto quale sia il valore assunto dalla variabile dipendente y quando la variabile indipendente x assume un particolare valore. Ad esempio:
se x = 1 allora y = 2 x = 2 * 1 = 2
se x = 2 allora y = 2 x = 2 * 2 = 4
se x = 3 allora y = 2 x = 2 * 3 = 6
se x = 10 allora y = 2 x = 2 * 10 = 20
e così via.

Una funzione matematica ad una variabile dipendente e ad una variabile indipendente può essere rappresentata mediante un *grafico* dove si riportano i punti corrispondenti alle coppie di valori relativi alla x ed ai corrispondenti valori della y. Ad esempio se vogliamo riportare il grafico relativo ai valori della nostra funzione appena calcolati abbiamo il grafico di Figura 1.
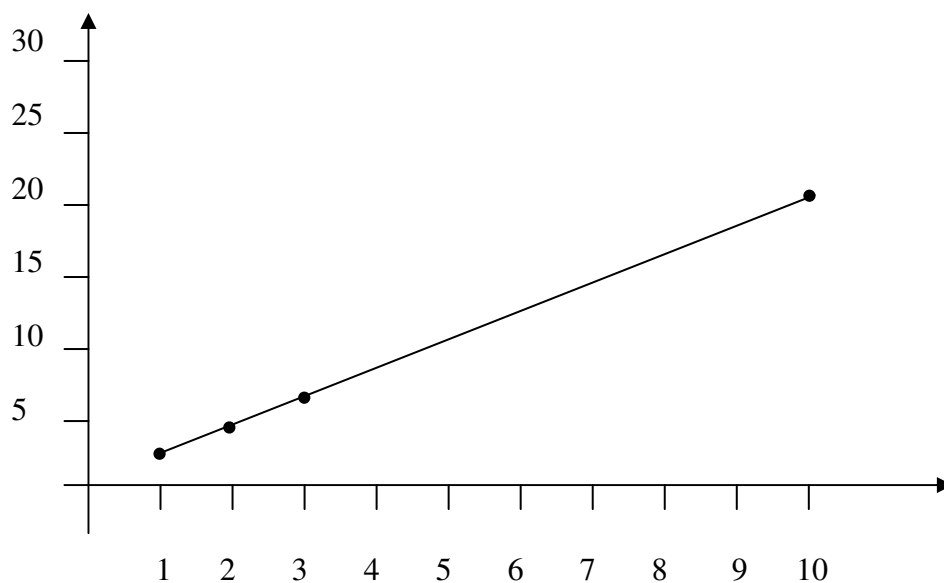
Figura 1 Il grafico di alcuni punti della funzione y = 2 x. Ogni pallino nero rappresenta una coppia di valori in cui il primo numero corrisponde alla x ed il secondo numero al relativo valore della y. Nel grafico è anche stata tracciata una linea che congiunge i punti che ci siamo calcolati per dare l'idea che la funzione rappresenta in realtà una linea continua.

Le funzioni matematiche vengono spesso indicate sinteticamente facendo riferimento con una semplice lettera alla particolare formula matematica che indica la relazione tra le variabili dipendenti e quelle indipendenti. Ad esempio le seguenti espressioni indicano due funzioni matematiche:
y = f(x)
y = g(x)
In queste dispense faremo uso della parentesi quadrate al posto delle tonde per indicare le funzioni, in modo che le parentesi utilizzate per indicare le funzioni non possano essere confuse con le parentesi che sono utilizzate in matematica per indicare l'ordine con cui debbono essere eseguite le operazioni (queste saranno delle parentesi tonde in questo scritto, ad esempio x * (y + z) indica che la somma va effettuata prima della moltiplicazione e non viceversa come accadrebbe normalmente).
Con le parentesi quadre i due esempi di funzione appena visti diventano:
y = f[x]
y = g[x]
mentre
y = f (x)
indica: "y è uguale ad f moltiplicato per x".

Gli "ingredienti" più usati per costruire le funzioni sono:
somma
sottrazione
divisione
moltiplicazione
potenza
radice
logaritmo

Vedremo ora le potenze e le radici poiché ci saranno utili tra poco, mentre non tratteremo i logaritmi.

La potenza è una espressione tipo:
2^5
In questa espressione 2 è detta *base* della potenza mentre 5 è detto *esponente* della potenza.

La potenza è un modo per indicare in modo sintetico una moltiplicazione tra numeri uguali. Ad esempio la moltiplicazione:
2 * 2
che implica moltiplicare due volte 2 per se stesso, si indica sinteticamente con le potenze come segue:
2^2
Invece:
2 * 2 * 2
poiché implica moltiplicare tre volte 2 per se stesso, si indica con:
2^3
e così via.

Nota che un numero elevato ad 1 è uguale al numero stesso. Invece un numero elevato a 0 viene posto pari ad 1 per convenzione.

La radice è un espressione del tipo:
$\sqrt[2]{9}$
dove 9 è detto *radicando* e 2 è detto *indice* della radice. La radice è definita come l'inverso della potenza. Questo significa che la radice con indice 2 del numero 9, indicata con:
$\sqrt[2]{9}$
vale 3. Infatti se si prende 3 (radice) e si eleva a potenza 2 (indice della radice), si ottiene 9 (radicando). Cioè si ha che:
$\sqrt[2]{9} = 3$ perché: $3^2 = 3 * 3 = 9$

Altri esempi:
$\sqrt[2]{0,25} = 0,5$ perché $0,5^2 = 0,5 * 0,5 = 0,25$
$\sqrt[3]{8} = 2$ perché $2^3 = 2 * 2 * 2 = 8$

Utilizzando la radice quadrata è anche possibile definire le potenze che hanno per *esponenti numeri con la virgola*, ad esempio:
$4^{0,5}$
Le potenze con esponenti con la virgola non sono delle nuove entità numeriche, sono solo un modo veloce di indicare potenze e radici insieme. Ad esempio per vedere a cosa corrisponde $4^{0,5}$ bisogna prima prendere l'esponente e trasformarlo in un rapporto matematico tra numeri interi. Nel nostro caso questa trasformazione porta a dire che 0,5 corrisponde al rapporto matematico 1/2:
0,5 = 1/2
Poi bisogna costruire un nuovo numero dove
- Si prende la base della potenza, cioè 4
- La si eleva a potenza con un esponente pari al numeratore del rapporto, nel nostro caso l'1 di 1/2. Nel nostro esempio si ottiene quindi
  4^1
- Poi si deve fare la radice di quanto ottenuto, cioè di 4^1, utilizzando per indice della radice il denominatore del rapporto, cioè il 2 nel caso del nostro 1/2. Nel nostro caso si ottiene quindi
  $\sqrt[2]{4^1}$

Ecco altri esempi di potenze con esponente con numeri con la virgola:

$4^{0,6666...} = 4^{2/3} = \sqrt[3]{4^2}$
$35,41265^{0,71428...} = 35,41265^{5/7} = \sqrt[7]{35,41265^5}$
(si noti che esiste un particolare algoritmo, che qui non presentiamo, che consente di calcolare quale sia il rapporto tra numeri interi che corrisponde ad un particolare numero con la virgola).

Le potenze con numeri con la virgola per esponente ci consentono di calcolare funzioni dalle forme più diverse, ad esempio:
$y = 2^x$
Se vogliamo tracciare il grafico di questa funzione possiamo ad esempio calcolarci dei valori di x e i corrispondenti valori della y (prova a farlo con la calcolatrice), e poi tracciare i punti su un grafico (v. Figura 2):

x	$y = 2^x$
0	1
0,5	1,414
1	2
1,5	2,828
2	4
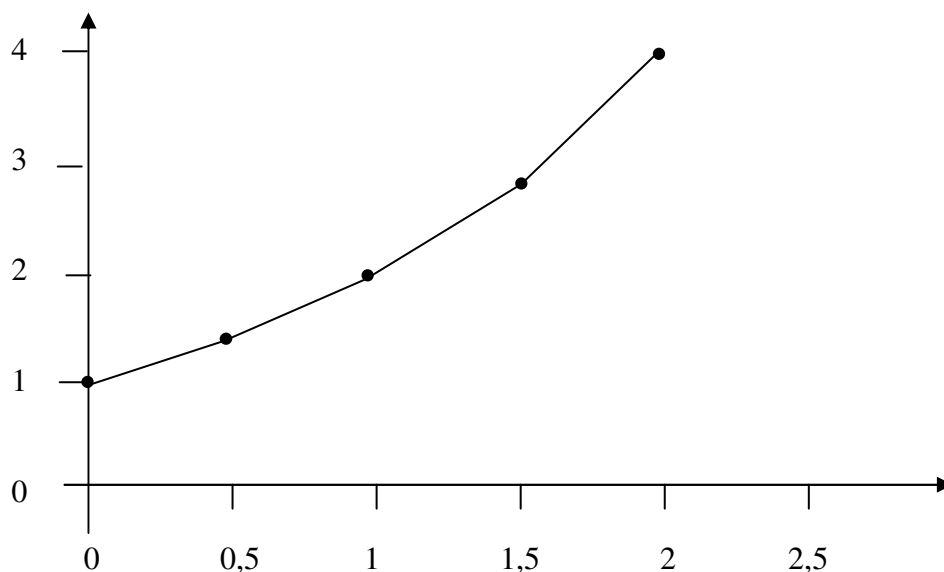
Figura 2 Il grafico di alcuni punti della funzione $y = 2^x$. Ogni pallino nero rappresenta una coppia di valori che ci siamo calcolati per la x e per i corrispondenti valori di y. Nel grafico è anche stata tracciata una linea che congiunge i punti calcolati.

Un ultima convenzione che viene utilizzata, è quella di indicare i rapporti utilizzando le potenze negative. Ad esempio secondo questa convenzione:
4^{-2}
equivale a:
$1 / (4^2)$
Mentre:
$123,123^{-456,456}$
equivale a:
$1 / (123,123^{456,456})$

Questa convenzione deriva dal fatto che le potenze ci consentono di trattare in modo semplice alcuni rapporti matematici. Ad esempio il rapporto:
$2^5 / 2^3 = (2 * 2 * 2 * 2 * 2) / (2 * 2 * 2) = 2 * 2 = 2^2$
può essere calcolato facilmente facendo direttamente riferimento ad operazioni tra esponenti:
$2^5 / 2^3 = 2^{5-3} = 2^2$

Viste queste proprietà degli esponenti delle potenze la convenzione sugli esponenti negativi risulta allora utile, come mostrato nel seguente esempio:
$2^3 / 2^5 = (2 * 2 * 2) / (2 * 2 * 2 * 2 * 2) = 1 / (2 * 2) = 1 / 2^2$
che utilizzando può anche essere calcolata più facilmente facendo uso degli esponenti negativi:
$2^3 / 2^5 = 2^{3-5} = 2^{-2} = 1 / 2^2$
(Questo giustifica anche la convenzione di porre pari ad 1 un numero elevato a 0: ciò viene fatto perché ad esempio $2^3/2^3 = 1$ ma nello stesso tempo $2^3/2^3 = 2^{3-3} = 2^0$ per cui è comodo porre $2^0 = 1$).

Dai neuroni alle reti neurali

Il modo migliore per capire cosa sia una rete neurale è iniziare a capire come in essa siano rappresentati i neuroni, i segnali scambiati dai, e le sinapsi. Un neurone è un componente di base del cervello che di solito possiede le caratteristiche indicate in Figura 3.
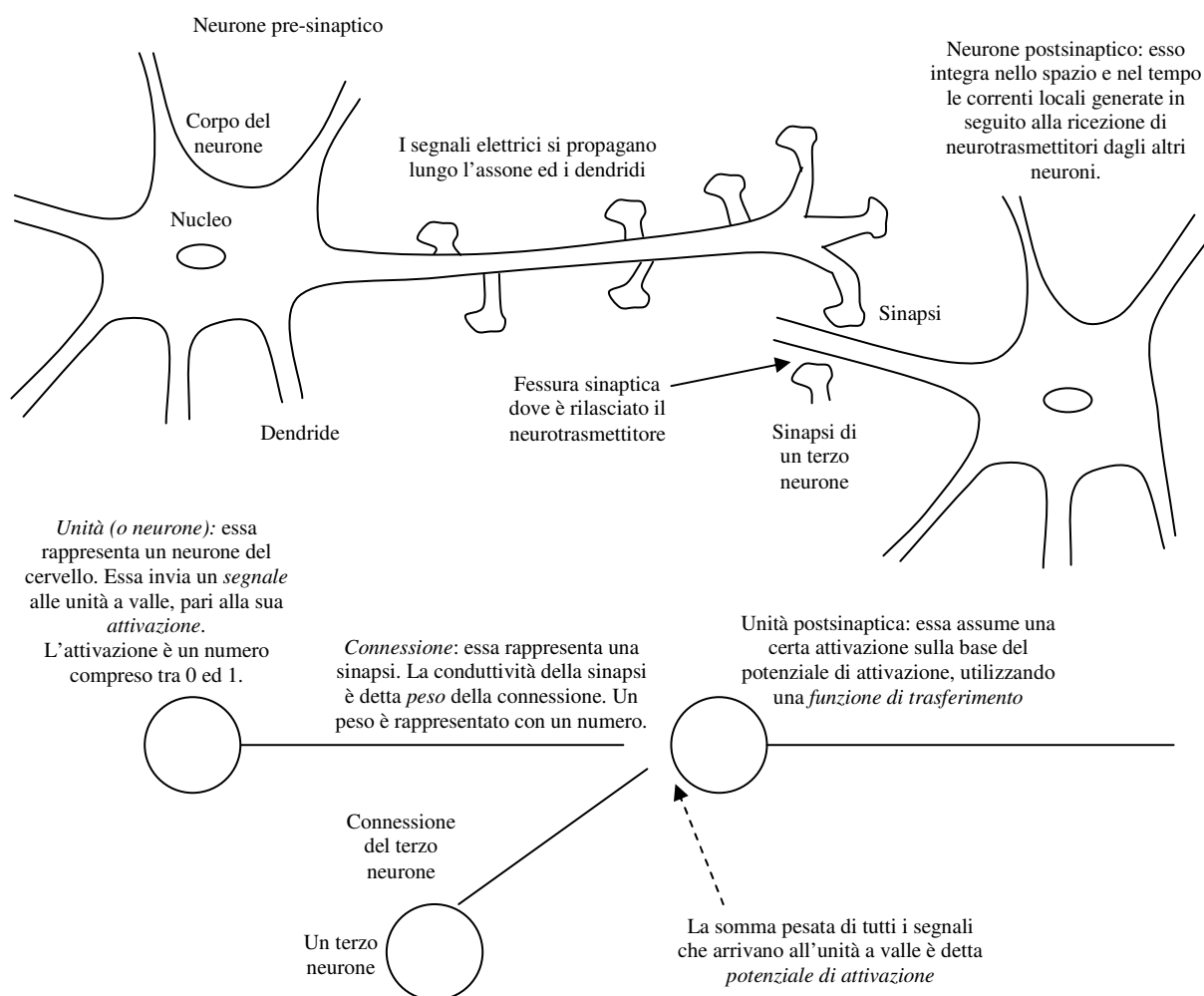
Figura 3 Un neurone con le tipiche parti costituenti, e modello con cui esso viene rappresentato nelle reti neurali.

Un *neurone* del cervello è una cellula composta da un *corpo*, da dei prolungamenti detti *dendridi*, e da un prolungamento maggiore detto *assone*. Dall'assone si dipartono una serie di piccole protuberanze dette *sinapsi*, che fungono da elementi di contatto con gli altri neuroni. I neuroni sono in grado di trasmettere un segnale elettrico lungo i loro assoni. Quando questo segnale elettrico arriva in prossimità delle sinapsi, le sinapsi rilasciano una certa quantità di sostanze chimiche, dette *neurotrasmettitori,* nel piccolo spazio che le separa dal dendride o dal corpo del neurone cui sono attaccate (*fessura sinaptica*). La quantità di neurotrasmettitore rilasciato nella fessura sinaptica costituisce in un certo senso la *conduttività della sinapsi*, cioè quanto la sinapsi esalta o attenua il segnale elettrico che arriva dall'assone. A valle della fessura sinaptica, il neurone è dotato di *recettori* in grado di catturare il neuromodulatore. Quando questo avviene, si generano delle correnti locali nei pressi della sinapsi. Queste correnti possono sommarsi nello spazio e nel tempo in prossimità dei dendridi e del neurone. Se la somma delle correnti che arriva presso la base dell'assone supera una certa soglia, si genera un impulso (detto *spike*) di corrente di un certo livello e di breve durata (2-5 millisecondi) che poi viaggia lungo l'assone stesso verso le sinapsi. Quando l'impulso arriva alle sinapsi queste rilasciano i neurotrasmettitori, ed il processo si ripete per i neuroni a valle.

In una rete neurale i neuroni sono rappresentati come delle *unità* in grado di elaborare in modo molto semplice, come vedremo tra poco, i segnali che ricevono in ingresso dagli altri neuroni. Le sinapsi sono rappresentate come delle *connessioni* tra le unità. Il segnale elettrico che viaggia lungo l'assone è rappresentato con un numero (di solito compreso tra 0 ed 1 nelle reti che vedremo). La conduttività delle sinapsi è rappresentata con un numero detto *peso della connessione*. L'effetto del peso sul segnale veicolato dalla connessione viene rappresentato *moltiplicando* il segnale stesso per il peso prima che esso raggiunga l'unità a valle. Ad esempio se x è il segnale che viaggia lungo la connessione tra un primo neurone ed un secondo neurone, e w è il peso di questa connessione, il segnale "pesato" che arriva al neurone a valle sarà pari a:

$w * x$

Facendo un esempio ancora più concreto, se x = 0,5 e w = 0.1, allora:

$w * x = 0,1 * 0,5 = 0,05$

mentre se x = 0,5 e w = 5, allora:

$w * x = 5 * 0,5 = 2,5$

Le unità di solito compiono due operazioni molto semplici sui segnali in arrivo: calcolano il *potenziale di attivazione* e si *attivano*.

Il *calcolo del potenziale di attivazione* avviene sommano i segnali pesati che arrivano dagli altri neuroni. Così ad esempio se:

x_1, x_2, x_3

sono i segnali che arrivano dagli altri neuroni attraverso le tre connessioni con i pesi

w_1, w_2, w_3

allora il potenziale di attivazione P si calcola come segue:

$P = w_1 x_1 + w_2 x_2 + w_3 x_3$

Le unità si *attivano* sulla base del potenziale di attivazione. Questo significa che esse mandano un certo segnale a tutte le unità collegate con esse a valle, sulla base del potenziale di attivazione ricevuto. Il calcolo del segnale da inviare alle altre unità sulla base del potenziale di attivazione viene fatto mediante l'utilizzo di funzioni matematiche tipiche, chiamate *funzioni di trasferimento*. Ci sono diversi tipi funzioni di trasferimento, che definiscono il tipo di unità che le utilizzano. Vedremo ora le più comuni.

Unità binaria a soglia
Le unità binarie a soglia possono attivarsi solo con il valore 0 o il valore 1. La funzione di trasferimento è tale che l'attivazione A dipende dal potenziale di attivazione P come segue:
$A = 0$ se $P < S$
$A = 1$ se $P \geq S$
S è un numero costante chiamato *soglia*, con cui il potenziale di attivazione viene confrontato per decidere l'output dell'unità. La legge di attivazione dell'unità a soglia in pratica dice che l'attivazione dell'unità sarà 0 se il potenziale di attivazione non supera la soglia, mentre sarà 1 se la supera. Questa funzione di attivazione riproduce un aspetto dei neuroni veri, caratterizzati dal fatto che lo spike del neurone viene prodotto solo se le correnti totali presenti nel neurone stesso superano un certo livello. La Figura 4 mostra come esempio il grafico della funzione di trasferimento a soglia nei casi in cui S sia pari a 0 o ad 1,5. Il grafico mostra sull'asse orizzontale i valori possibili del potenziale di attivazione e sull'asse verticale i corrispondenti valori di attivazione dell'unità.
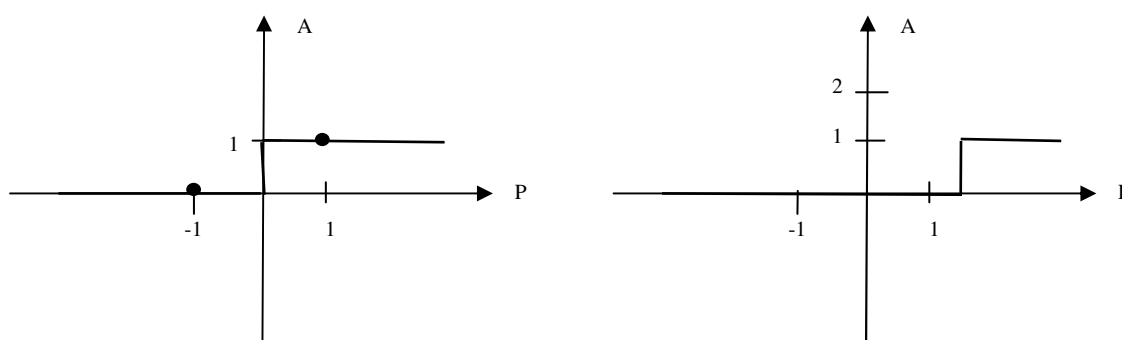
Figura 4 Sinistra: Funzione di trasferimento a soglia con soglia pari a 0. I pallini evidenziano due esempi di valori pari a 0 ed 1 assunti dall'attivazione in corrispondenza di potenziali di attivazione pari a –1 o 1. Destra: Funzione di trasferimento a soglia con soglia pari a 1,5.

Unità booleane

Le unità booleane sono delle unità simili alle porte logiche di un computer, che operano sulla base di segnali binari, 0 o 1. In effetti le unità booleane caratterizzate dall'avere attivazioni discrete pari o a 0 o ad 1. Inoltre i pesi delle connessioni di queste unità sono sempre pari ad 1. L'aspetto interessante delle unità booleane è la funzione di trasferimento, che può essere di vari tipi, i più comuni dei quali sono detti: OR, AND, NOT, X-OR (o OR esclusivo). Queste funzioni di trasferimento possono essere descritte con delle tabelle, visto che il numero di possibili combinazioni di segnali in input sono in numero limitato (ad esempio una unità che riceve segnali da sole altre due unità può ricevere in input solo le combinazioni di valori: 0 0; 0 1; 1 0; 1 1).

La funzione di trasferimento delle *unità booleane OR* è tale che l'attivazione dell'unità è pari ad 1se e solo se almeno uno dei segnali in input è pari ad 1. Nel caso di due segnali in input la funzione di trasferimento OR può quindi essere sintetizzata dalla seguente tabella:

Input 1	**Input 2**	**Output**
0	0	0
0	1	1
1	0	1
1	1	1

La funzione di trasferimento delle *unità booleane AND* è tale che l'attivazione dell'unità è pari ad 1 se e solo se tutti i segnali in input sono pari ad 1. Nel caso di due segnali in input la funzione di trasferimento AND può quindi essere sintetizzata dalla seguente tabella:

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

La funzione di trasferimento delle *unità booleane X-OR* è tale che l'attivazione dell'unità è pari ad 1 se almeno dei segnali in input è pari ad 1, mentre vale zero nel caso in cui tutti i segnali in input sono o tutti pari a 0 o tutti pari a 1. Nel caso di due segnali in input la funzione di trasferimento X-OR può quindi essere sintetizzata dalla seguente tabella (questa funzione si chiama *OR esclusivo* in quanto è simile all'OR con l'eccezione del caso in cui tutti i segnali in input sono pari ad 1):

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

La funzione di trasferimento dell'*unità booleana NOT* può essere applicata solo ad unità con un solo segnale di input. Questa funzione di trasferimento sostanzialmente "inverte" il segnale in input. La funzione di trasferimento NOT può quindi essere sintetizzata dalla seguente tabella:

Input	Output
0	1
1	0

Unità lineari
Una unità lineare è una unità con una funzione di trasferimento che produce in output il potenziale di attivazione stesso, senza modificarlo. L'attivazione A del neurone è quindi uguale al potenziale di attivazione P:
A = P
Il grafico di questa funzione di trasferimento è appunto una linea (Figura 5).
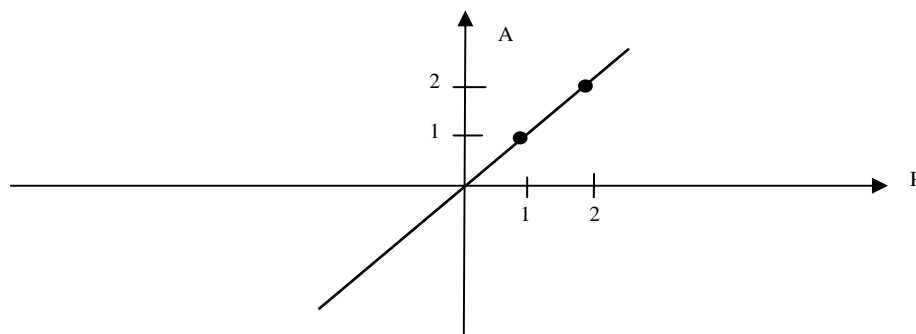
Figura 5 Funzione di trasferimento lineare.

Unità sigmoidali
Una unità lineare è una unità con una funzione di trasferimento che produce in output una attivazione A che è una funzione abbastanza complessa del potenziale di attivazione P. Questa funzione è detta sigmoide, ed ha la seguente equazione:

$$A = \frac{1}{1 + e^{-P}} = \frac{1}{1 + \frac{1}{e^{P}}}$$

In questo equazione e è il numero di Nepero, un particolare numero utilizzato in analisi matematica ed avente una serie di proprietà utili sulle quali però non possiamo soffermarci. Il numero di Nepero vale circa 2,718281828. Il grafico della funzione sigmoide è una versione senza discontinuità della funzione binaria (Figura 6).
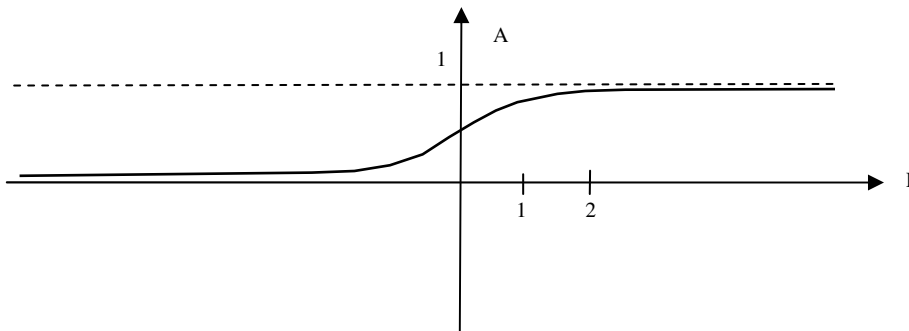
Figura 6 Funzione di trasferimento sigmoide.

Il perceptrone di Rosenblatt e la Regola Delta di apprendimento

Ora esamineremo una delle prime reti utilizzate al fine di modellizzare dei processi psicologici, il perceptrone di *Rosenblatt (1958)*. Il perceptrone può essere utilizzato ad esempio per mimare la categorizzazione di immagini in due categorie, 0 ed 1. Ad esempio può essere utilizzato per il riconoscimento dell'immagine della lettera "a" tra altre lettere. Il perceptrone è una rete neurale costituita da uno strato di unità di input ed uno strato di unità di output. L'architettura di un perceptrone con una sola unità di output è illustrato in Figura 7. Questo può ad esempio essere utilizzato per effettuare il riconoscimento di un particolare pattern di input tra altri pattern di input, nel senso che gli si può chiedere di apprendere ad emettere un valore di output pari ad 1 nel caso in cui in input sia presentato il pattern da riconoscere, e 0 nel caso in cui in input sia presentato un altro pattern. L'unità di uscita è una unità binaria a soglia con soglia pari a 0. In realtà il neurone può avere una soglia implementata con un una "unità di bias". Questo modo di rappresentare la soglia consente alla rete di apprendere la soglia corretta al pari di degli altri pesi. Per semplicità espositiva, la trattazione del bias è rinviata a dopo.

Le reti neurali sono caratterizzate da due processi importanti. Il primo processo è il processo di attivazione attraverso il quale la rete riceve dei segnali in input e, mediante la propagazione di questi segnali lungo le connessioni e le funzioni di trasferimento delle unità, arriva a produrre una certa risposta in output. Il secondo processo è il processo di apprendimento. Durante l'apprendimento i pesi delle connessioni della rete sono modificate opportunamente così che la rete risponda nel modo desiderato agli input.

Iniziamo a vedere il *processo di attivazione del perceptrone*. Indichiamo con x_i l'attivazione dell'unità di input i-esima, indichiamo con P il potenziale di attivazione dell'unità di uscita, ed infine indichiamo con A l'attivazione di questa unità. Un pattern di input è una serie di numeri, in numero pari alle unità di input, che indicano come si attivano le unità di input della rete. Un pattern di input può essere ad esempio il seguente (per semplicità consideriamo un pattern a valori binari, ma il perceptrone funziona anche con pattern di input formati da valori intermedi tra 0 ed 1):
$x_1 = 1$, $x_2 = 1$, $x_3 = 0$, $x_4 = 0$, $x_5 = 1$

Se indichiamo con w_i il peso relativo alla i-esima connessione, si ha che il potenziale di attivazione è pari alla somma dei segnali in input ciascuno moltiplicato per il suo peso:
$P = x_1 * w_1 + x_2 * w_2 + x_3 * w_3 + x_4 * w_4 + x_5 * w_5$

Secondo la funzione di attivazione binaria a soglia con soglia 0 si ha che:
$A = 0$ se $P < 0$
$A = 1$ se $P \geq 0$
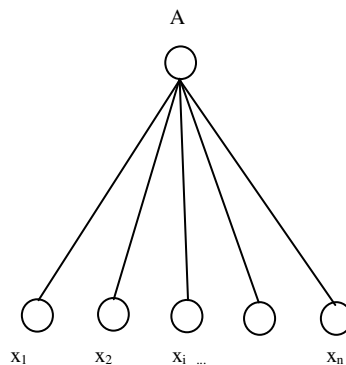
Figura 7 L'architettura della rete neurale perceptrone.

Vediamo ora il processo di apprendimento del perceptrone. Il perceptrone viene addestrato con una *procedura di addestramento supervisionata*. Questo significa che il perceptrone deve imparare ad associare una certa risposta, detta *output desiderato*, a diversi possibili pattern di intput. Supponiamo che i pattern in input, con le risposte desiderate che perceptrone deve dare, siano rappresentate dai seguenti pattern:

Pattern di input	**Output desiderato**
1 1 0 0 1	1
1 0 1 0 1	1
0 1 0 0 0	0
0 1 0 1 0	0
0 1 1 1 0	0

Inizialmente vengono assegnati dei pesi casuali al perceptrone, ad esempio dei valori scelti a caso tra –1 ed 1 tipo:
-0,1 0,9 -0,4 -0,7 -0,1

I pattern di input sono presentati uno ad uno al perceptrone, dal primo all'ultimo, per diverse volte, finché il perceptrone non risponde nel modo desiderato a tutti i pattern di input. Ogni volta che un pattern di input viene presentato al perceptrone, si propagano i segnali attraverso le connessioni della rete neurale, si calcola il potenziale di attivazione P dell'unità di output, e poi si calcola l'attivazione di essa, e questa viene interpretata come risposta data. Poi si modificano i pesi sulla base della risposta data A e della risposta desiderata A', secondo quanto stabilito dalle seguenti regole:

Se A = 0 e A' = 0, oppure A = 1 e A' = 1
allora i pesi sono lasciati come sono.

Se A = 1 ed A’ = 0, oppure se A = 0 ed A’ = 1
allora i pesi sono cambiati come segue:
w_i(nuovo) = w_i(vecchio) + (η * (A’-A) * x_i)

Il senso di queste regole può essere spiegato come segue.
η è un coefficiente positivo scelto intorno ad 1, detto *coefficiente di apprendimento*; esso serve per regolare la velocità di apprendimento, cioè di quanto sono modificati i pesi.
La prima regola fa in modo che se la risposta data dal perceptrone è come quella desiderata, allora i pesi non sono modificati.
La terza regola fa in modo che se il valore desiderato A’ è più alto del valore A restituito dalla rete, allora il peso viene alzato (infatti A’-A è positivo, vedi formula).
Invece se il valore desiderato A’ è più basso del valore A restituito dalla rete, allora il peso viene abbassato (infatti A’-A è negativo, vedi formula).
La modifica apportata ai pesi viene moltiplicata per x_i perché si vogliono modificare soprattutto i pesi corrispondenti alle unità di input che hanno contribuito maggiormente a dare la risposta data, cioè le unità di input con una attivazione grande. Ad esempio nel nostro caso, visto che le unità di input si attivano o con 0 o con 1, di fatto la regola implica che sono modificati solo i pesi corrispondenti alle unità di input pari ad 1. Infatti per tutte le altre la modifica dei pesi sarà pari a:
(η * (A’-A) * 0) = 0.

Proviamo ora a vedere come vengono modificati i pesi della rete indicata sopra con i pattern di apprendimento visti sopra, ponendo η pari a 0,5:

Pattern di input	Pesi	P	A	A’	Pesi modificati
1 1 0 0 1	-0,1 0,9 -0,4 -0,7 -0,1	0,7	1	1	-0,1 0,9 -0,4 -0,7 -0,1
1 0 1 0 1	-0,1 0,9 -0,4 -0,7 -0,1	-0,6	0	1	0,4 0,9 0,1 -0,7 0,4
0 1 0 0 0	0,4 0,9 0,1 -0,7 0,4	0,9	1	0	0,4 0,4 0,1 -0,7 0,4
0 1 0 1 0	0,4 0,4 0,1 -0,7 0,4	-0,3	0	0	0,4 0,4 0,1 -0,7 0,4
0 1 1 1 0	0,4 0,4 0,1 -0,7 0,4	-0,2	0	0	0,4 0,4 0,1 -0,7 0,4

Visto che il perceptrone ha fatto due errori, procediamo ad un secondo ciclo di apprendimento:

Pattern di input	Pesi	P	A	A’	Pesi modificati
1 1 0 0 1	0,4 0,4 0,1 -0,7 0,4	1,2	1	1	0,4 0,4 0,1 -0,7 0,4
1 0 1 0 1	0,4 0,4 0,1 -0,7 0,4	0,9	1	1	0,4 0,4 0,1 -0,7 0,4
0 1 0 0 0	0,4 0,4 0,1 -0,7 0,4	0,4	1	0	0,4 -0,1 0,1 -0,7 0,4
0 1 0 1 0	0,4 -0,1 0,1 -0,7 0,4	-0,8	0	0	0,4 -0,1 0,1 -0,7 0,4
0 1 1 1 0	0,4 -0,1 0,1 -0,7 0,4	-0,7	0	0	0,4 -0,1 0,1 -0,7 0,4

Visto che il perceptrone ha fatto ancora un errore, procediamo ad un terzo ciclo di apprendimento:

Pattern di input	Pesi	P	A	A’	Pesi modificati
1 1 0 0 1	0,4 -0,1 0,1 -0,7 0,4	0,7	1	1	0,4 -0,1 0,1 -0,7 0,4
1 0 1 0 1	0,4 -0,1 0,1 -0,7 0,4	0,9	1	1	0,4 -0,1 0,1 -0,7 0,4
0 1 0 0 0	0,4 -0,1 0,1 -0,7 0,4	-0,1	0	0	0,4 -0,1 0,1 -0,7 0,4
0 1 0 1 0	0,4 -0,1 0,1 -0,7 0,4	-1,1	0	0	0,4 -0,1 0,1 -0,7 0,4
0 1 1 1 0	0,4 -0,1 0,1 -0,7 0,4	-1,0	0	0	0,4 -0,1 0,1 -0,7 0,4

Visto che il perceptrone non ha fatto alcun errore, l’apprendimento termina.

Il perceptrone ha diversi aspetti interessanti, condivisi con la maggior parte delle reti neurali. Ora saranno illustrati brevemente.

1. Il perceptrone da luogo ad un processamento delle informazioni in parallelo, nel senso che il pattern di input attiva contemporaneamente tutte le unità di input ed i segnali attraversano le coesioni nello stesso momento, proprio come avviene per i neuroni e le sinapsi del cervello.
2. Nel perceptrone le *informazioni sono immagazzinate in modo distribuito nei vari pesi*. Infatti ogni pattern viene riconosciuto utilizzando tutti i pesi, ed ogni peso partecipa alla classificazione di tutti i pattern.
3. Il perceptrone sa rispondere anche a dei pattern che non ha mai visto prima. In questo caso esso tenderà a dare una risposta sulla base della somiglianza di questi pattern con i pattern già visti durante l'addestramento. Ad esempio con il pattern 0 0 1 1 0, pattern che il perceptrone non ha mai visto prima ma che somiglia all'ultimo pattern del set di apprendimento, esso risponde 0, così come aveva imparato a rispondere all'ultimo pattern. Questa importante proprietà delle reti neurali viene detta proprietà di *generalizzazione*.
4. Il perceptrone, mediante il processo di *apprendimento*, acquisisce una competenza "cognitiva" (ad esempio quella di classificare i pattern in input in due categorie, come nell'esempio visto). Questa capacità è acquisita progressivamente con l'esperienza modificando progressivamente i pesi.
5. Il perceptrone presenta una notevole *robustezza rispetto al rumore*. Si provi ad esempio a scegliere un 1 o uno 0 a caso per ogni pattern, ed a scambiali con un uno 0 e con un 1 rispettivamente. Si provi poi a far rispondere la rete neurale a questi pattern "danneggiati": si potrà verificare che il perceptrone continuerà a dare una risposta giusta al maggior numero dei pattern.
6. Le reti neurali sono caratterizzate dal *degradamento grazioso.* Per vedere cosa significa questo, si provi ad eliminare una unità di input scelta a caso del perceptrone e la sua connessione. Si provi poi a far funzionare il perceptrone con i pattern di input soliti e le unità di input rimanenti (ad esempio, se si elimina l'ultima unità del perceptrone, il primo pattern di input sarà presentato alle prime quattro unità del perceptrone nella forma 1 1 0 0). Man mano che si eliminano altre unità, la performance del perceptrone peggiorerà gradualmente. Questo fenomeno viene appunto detto degradamento grazioso e mostra che le reti neurali hanno la robustezza tipica dei sistemi cerebrali naturali e che mantengono gran parte delle loro funzionalità anche perdendo parte della loro struttura. Questa proprietà non è invece posseduta dagli altri sistemi di intelligenza artificiale che tendono a perdere completamente il loro funzionamento non appena si distrugge anche solo una piccola porzione di essi.

Cenni ad altri tipi di rete

Esistono molti altri tipi di reti neurali. Ad esempio il perceptrone stesso può essere dotato di più di una unità di uscita. Questo può consentirgli di dividere gli input in più categorie oppure di imparare ad associare un particolare pattern di output ad ogni particolare pattern di input (Figura 8). Il funzionamento di questa versione modificata del perceptrone è tale che per ogni unità di output viene calcolato il potenziale di attivazione e l'attivazione in modo identico a quanto visto per l'unica unità di uscita del perceptrone ad una uscita già visto. Queste reti sono anche dette *reti feed-forward* ("feedforward" significa "a diffusione in avanti") in quanto in esse il segnale si propaga dalle unità di input alle unità di output ma non viceversa (anche il perceptrone è in realtà una rete feed-forward).
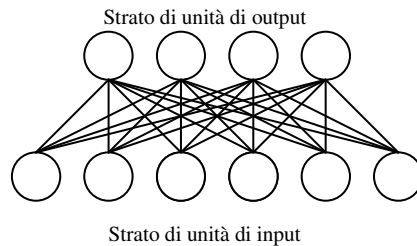
Figura 8 L'architettura del perceptrone a più uscite.

Una variante del *perceptrone* è una rete neurale che ha una *unità di uscita a sigmoide*. In questo caso il funzionamento della rete prevede il calcolo del potenziale di attivazione dell'unità a sigmoide, e poi l'utilizzo della funzione di trasferimento sigmoide per l'attivazione dell'unità. La procedura di apprendimento è leggermente diversa dalla regola delta, e la risposta che la rete da in uscita è un valore compreso tra 0 ed 1. La regola di apprendimento utilizzata per questo particolare tipo di perceptrone viene detta *regola di apprendimento Widrow-Off* dal nome degli autori che l'hanno proposta. Questa versione del perceptrone è adatta a risolvere problemi dove la risposta della rete neurale deve essere continua e non binaria. Ad esempio una rete di questo genere dotata di *due unità di output* può essere utilizzata per controllare i motori delle due ruote di cui è dotato un robot, in modo da far muovere le ruote più o meno velocemente a seconda dell'attivazione delle unità di uscita. In questo caso l'attivazione dei sensori del robot, ad esempio l'attivazione dei pixel (singoli puntini) della sua telecamera, sarebbe utilizzata per attivare istante per istante le unità di input della rete neurale. I segnali si propagherebbero così lungo le connessioni raggiungendo le unità di uscita e quindi i motori delle ruote. I pesi della rete neurale dovrebbero essere tali che, sulla base dell'immagine percepita dal robot, la rete neurale invierebbe comandi adeguati alle ruote.
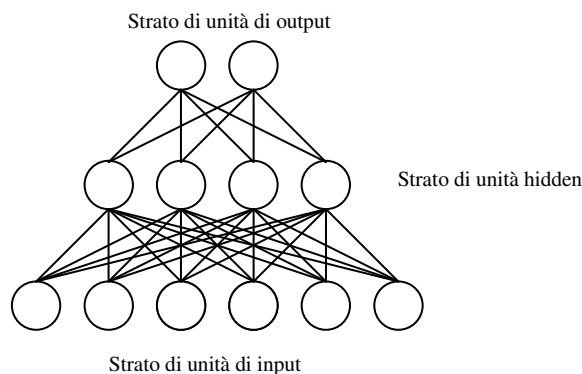
Figura 9 L'architettura di una rete neurale feed-forward a tre strati.

I perceptroni hanno una forte limitazione. Possono solo risolvere solo una classe di problemi detti "problemi linearmente separabili" (intuitivamente questo significa che i perceptroni possono imparare a classificare correttamente i pattern di input solo se questi pattern che formano diverse categorie si differenziano tra loro sulla base di determinati criteri). Per risolvere questo problema sono state proposte delle reti neurali a tre strati, dove tra lo strato di unità di input e quello di output vi è uno strato chiamato di unità chiamate unità hidden (cioè nascoste). Le *reti feed-forward con almeno tre strati* sono delle reti neurali che possono imparare ad associare qualsiasi pattern di output ai pattern di input, e sono le reti neurali più utilizzate. *Rumelhart, Hinton e Williams (1986)* hanno proposto un algoritmo di apprendimento molto potente, chiamato *Error-Back Propagation*, con cui addestrare queste reti. Intuitivamente, questo algoritmo è basato sul calcolo dell'errore sulle unità di uscita basato sul pattern di output desiderato (similmente a quanto abbiamo visto per il perceptrone). Questo errore è poi "propagato all'indietro" lungo le connessioni tra lo strato di unità

di output e quello hidden, e poi lungo le connessioni tra lo strato di unità hidden e quelle di input, in modo da modificare i pesi relativi a queste connessioni.
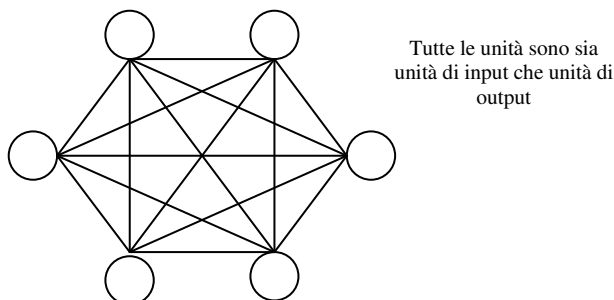
Figura 10 L'architettura della rete neurale di Hopfield.

I perceptroni e le altre reti feed-forward si basano *sull'apprendimento supervisionato*. L' apprendimento è detto supervisionato perché, come abbiamo visto nel caso del perceptrone, esso si basa sulla conoscenza del pattern di output desiderato, che si presume fornito da un *supervisore* del processo di apprendimento. Sono state proposte anche altre reti che apprendono sulla base di principi diversi dall'apprendimento supervisionato. Una classe importante di reti neurali si basa sull'*auto-organizzazione* delle reti stesse. Un primo tipo di reti di questo genere sono le *reti di Hopfield* (Hopfield, 1981). Esse comportano la memorizzare dei pattern che vengono presentati in input, e consentono di richiamare questi pattern sulla base della presentazione di *porzioni di essi*. Si tratta quindi di reti molto adatte a rappresentare i processi associativi della memoria. Le reti di Hopfield hanno una architettura fatta da un unico strato di unità tutte connesse tra loro da connessioni che vanno in entrambe le direzioni (Figura 10). Queste reti apprendono sulla base della famosa legge di Hebb. Questa legge dice che se due unità vengono attivate contestualmente, la connessione tra esse si rafforza. Questo comporta che se in un secondo momento una di queste unità viene attivata, il segnale che essa manda all'altra unità attraverso la connessione rafforzata, tende a far attivare anche questa seconda unità (quindi il pattern memorizzato all'inizio, viene richiamato attivando solo una porzione di esso).
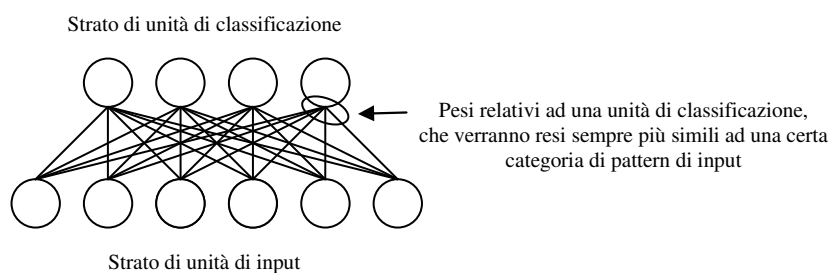
Figura 11 L'architettura della rete di Kohonen.

Un secondo tipo di reti ad auto-organizzazione sono le reti di Kohonen. Queste reti sono costituite da uno strato di input e da uno strato di output (Figura 11). Ogni unità di input è collegata a tutte le unità di output. Quando un pattern di input viene presentato alla rete, la rete confronta questo pattern con i pesi di ciascuna unità di uscita. L'unità di uscita i cui pesi somigliano di più al pattern viene detta *unità vincente*. La competizione tra le diverse unità viene detta *winner takes all* , cioè "il vincitore vince tutto", perché si immagina che le unità diano luogo ad una competizione per attivarsi, ed alla fine solo una vinca la competizione e si attivi. I pesi dell'unità vincitrice sono modificati in modo da essere resi ancora più simili a quelli del pattern di input. Con questo tipo di

apprendimento, le varie unità di output tendono ad attivarsi molto in corrispondenza di classi di pattern di input diverse. Questo significa che l'attivazione di una certa unità di uscita classifica il pattern di input sotto la categoria rappresentata da essa. Queste reti possono quindi essere utilizzate per simulare alcune parti del cervello dove avviene la classificazione dei pattern di ingresso in categorie diverse.

Usiamo Excel per costruire un semplice perceptrone con pesi fissati a mano

Costruiamo ora un semplice perceptrone con tre unità di input ed una unità di uscita sigmoidale, utilizzando Excel. Fisseremo i suoi pesi a mano, quindi non utilizzeremo alcun algoritmo di apprendimento. Mostreremo quindi come il perceptrone può implementare alcune funzioni logiche e come esso può essere utilizzato come modello quantitativo di alcune semplici funzioni cognitive.

Per prima cosa dobbiamo costruire l'architettura del perceptrone. Il grafico di Figura 12 mostra l'architettura che dobbiamo costruire.
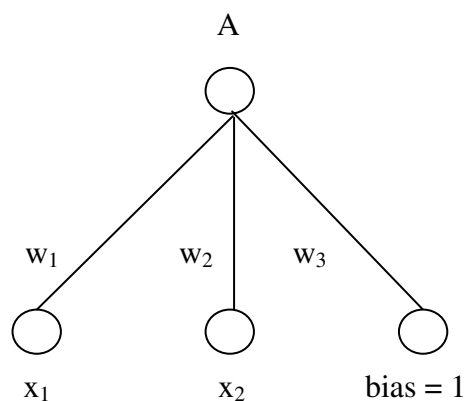
Figura 12 L'architettura del perceptrone che dobbiamo costruire.

Questa architettura presenta due unità di input, x_1 ed x_2, ed una unità di input particolare, chiamata bias, su cui ora ci soffermiamo. L'unità di bias è una unità speciale che si usa nelle reti che hanno delle unità sigmoidali, e quando è presente è di solito collegata a ciascuna di esse. Il bias si attiva costantemente con 1, cioè invia un segnale di 1 a tutti i neuroni a cui è connessa ogni volta che la rete funziona. A parte queste particolarità, l'unità di bias si comporta come tutte le altre unità. Ad esempio se si utilizza un algoritmo di apprendimento che modifica i pesi della rete neurale, i pesi relativi alle connessioni del bias saranno modificati nello stesso modo in cui sono modificati tutti gli altri pesi.

A che serve il bias? Il bias ha la funzione di creare una soglia modificabile per ciascuna unità sigmoidale a cui è collegato. Questa soglia è costituita dal peso della connessione con cui il bias è collegato alla particolare unità sigmoidale. Vediamo il perché di questo considerando una particolare unità sigmoidale a cui il bias è collegato attraverso una connessione con peso 0,5. L'unità di bias si attiva costantemente con 1 ed invia quindi un segnale pesato costante di 0,5 al neurone sigmoidale, indipendentemente da quali siano gli altri segnali che arrivano a questo neurone dagli altri neuroni.

Quel è la funzione della soglia costituita dal bias? La funzione è simile alla funzione svolta dalla soglia nelle unità binarie a soglia: fare in modo che il neurone si attivi con 1 solo se il totale dei

segnali pesati provenienti dalle altre unità è (sufficientemente) maggiore della soglia (presa con il segno meno). Questi aspetti risulteranno comunque più chiari con gli esempi mostrate tra poco.

Iniziamo quindi a costruire la rete neurale. L'obiettivo è predisporre delle celle per ospitare i pattern di input della rete, delle celle per ospitare i pesi delle connessioni, e delle celle contenenti delle *formule* per calcolare il potenziale di attivazione e l'attivazione dell'unità di output. Utilizzeremo poi questo insieme di elementi per fissare dei pesi delle connessioni della rete e fare quindi degli esperimenti dove invieremo dei pattern di input alla rete e valuteremo come essa risponderà.

Iniziamo con il riempire il foglio di lavoro con le etichette ed i valori numerici illustrati in Figura 13. Nota che il modo con cui ordineremo le varie parti della rete nel foglio di lavoro è solo uno di molti possibili. Le etichette ci servono per ricordare il contenuto delle varie celle. Nelle celle B2 e B3 metti provvisoriamente i valori 0 e 0. Queste celle ospitano i valori corrispondenti al pattern di input a cui la rete deve rispondere. Essi saranno modificati quando faremo degli esperimenti per vedere come la rete risponde ad essi. Nella cella B4 mettiamo 1: questa è l'attivazione dell'unità di bias e non verrà più modificata. Nelle celle C2, C3, C4 mettiamo dei pesi provvisori come in figura, relativi alle tre connessioni rispettivamente corrispondenti alla prima unità di input, alla seconda unità di input ed al bias. Questi pesi saranno modificati quando testeremo la rete (nota che anche il peso del bias sarà modificato negli esperimenti). Ora riempiamo le altre celle:

- La colonna D ospita i segnali pesati provenienti dalle tre unità di input. In D2 scrivi la formula: "=B2*C2". Seleziona poi la cella e ricopia la formula nelle celle D2 e D3 trascinando su esse il quadratino di riempimento (Figura 14).
- Ora inseriamo in E2 la formula per calcolare il potenziale di attivazione: "=D2+D3+D4".
- Poi inseriamo in F2 la formula per calcolare l'attivazione dell'unità sigmoidale di uscita sulla base del potenziale di attivazione: "=1/(1+EXP(-E2))". A questo punto il foglio di lavoro dovrebbe apparire con in Figura 15.

Nota che in questa ultima formula facciamo uso della funzione di Excel chiamata "exp()" che serve per calcolare la potenza in base "e" (numero di Nepero) di un esponente indicato da noi. Nel nostro caso l'esponente usato è il potenziale di attivazione contenuto in E2, preso con il segno meno. Ora la rete è completa e possiamo utilizzarla per fare degli esperimenti.

Disegno ▾ Forme ▾

G20 =

	A	B	C	D	E	F	G
1		Pattern di input xi	Pesi wi	xi*wi	Pote di atti P	Attivazione A	
2	Unità input 1	0	20				
3	Unità input 1	0	20				
4	Bias	1	-10				
5							

Figura 13 Ecco come appare il foglio di lavoro dopo che abbiamo inserito i primi elementi della rete.
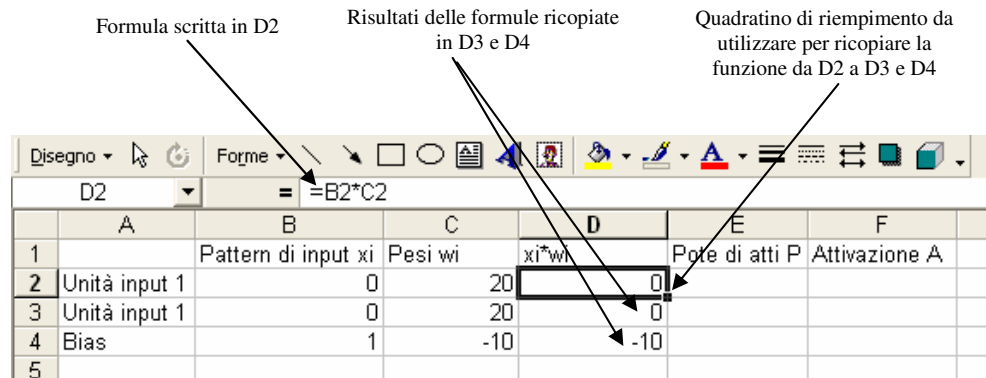
Figura 14 Ecco come appare il foglio di lavoro dopo che abbiamo inserito altri elementi della rete.
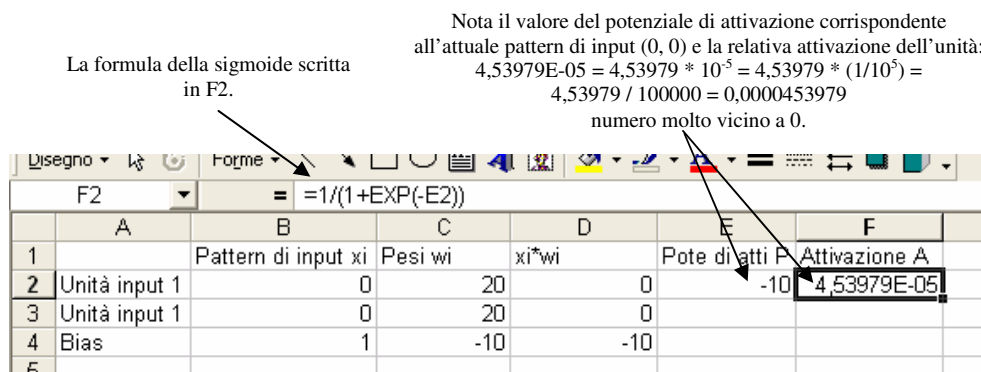
Figura 15 Ecco come appare il foglio di lavoro dopo che abbiamo inserito tutti gli elementi della rete.

Primo esperimento: implementazione di unità booleane OR ed AND

Nel nostro primo esperimento mostriamo che la nostra rete può essere utilizzata per implementare le unità booleane OR ed AND che abbiamo visto prima. Anche se nelle reti booleane le unità booleane sono utilizzate come unità di base per costruire l'intera rete, in realtà esse effettuano dei calcoli piuttosto complessi. Mostrando che esse possono essere implementate con il nostro piccolo perceptrone intendiamo mostrare un concetto generale: mettendo opportunamente insieme più unità molto semplici, possiamo ottenere delle funzionalità molto complesse. Questo dovrebbe dare una idea intuitiva di come sia possibile che il comportamento umano ed animale, pur così complesso, possa essere prodotto un numero sufficientemente grande di unità elementari come i neuroni.

Cominciamo con il cercare i pesi per implementare l'unità OR. Una unità OR deve essere tale da attivarsi con 1 se almeno una delle unità di input si attivano con 1. Per ottenere questo possiamo settare i pesi della rete come indicato in Figura 15. Questi pesi hanno questo effetto.
Quando le unità di input si attivano con (0, 0), allora il bias invierà un segnale pesato di –10 all'unità, mentre le altre due invieranno un segnale pesato di 0 (vedi valori nella colonna D della figura). La sigmoide allora causerà un'attivazione dell'unità di uscita della rete vicina a 0 (vedi valore nella cella F2 in figura) come desiderato sulla base della funzione OR.
Prova invece ad inserire le coppie di valori (1, 0) o (0, 1) nelle celle che ospitano il pattern di input, B2 e B3. Noterai che l'unità attiva con 1 invierà un segnale pesato di +20 all'unità di uscita. Questo valore sopravanzerà di +10 il segnale pesato del bias (-10). Sulla base di questo potenziale di attivazione (+10) la sigmoide provocherà poi una attivazione vicina ad 1 dell'unità di uscita, come desiderato sulla base della funzione OR.

Prova infine ad inserire l'ultimo pattern di input, (1, 1), nelle celle B2 e B3. Ora entrambe le unità manderanno un segnale pesato di +20 e +20 all'unità di input, per cui il potenziale di attivazione sarà +30 ed in base alla sigmoide l'unità di uscita si attiverà con un valore vicino ad 1, come desiderato sulla base della funzione OR.

Vediamo ora come implementare la funzione AND. Sulla base di questa funzione, la rete deve restituire 1 solo se entrambe le unità di input sono pari ad 1, mentre deve restituire 0 negli altri tre casi. Prova tu a trovare una combinazione di pesi opportuni per avere la funzione AND prima di procedere oltre.
Una possibile soluzione del problema è basata sui pesi (+20, +20, -30). Questa soluzione funziona perché quando entrambe le unità di input sono pari a 0, il potenziale di attivazione varrà –30 a causa del segnale pesato del bias, così che la rete restituirà 0.
Nel caso in cui una sola unità di input si attiva con 1, il suo segnale pesato, pari a +20, non riesce a superare la soglia del bias, -30, per cui il potenziale di attivazione è –10 e l'unità di uscita si attiva con 0.
Nel caso invece in cui entrambe le unità di input valgono 1, allora la somma dei loro due segnali pesati da +40 e riesce a superare la soglia, -30, così da causare l'attivazione di 1 dell'unità di uscita.

Secondo esperimento: (piccola) invarianza percettiva e "matching to target"

I due perceptroni appena visti, che implementano le unità OR ed AND, possono anche essere interpretati come due modelli di processi psicologici percettivi e decisionali. Ad esempio il perceptrone dell'OR può costituire il modello di una porzione del sistema visivo che si attiva quando una certa "feature" di una immagine è presente in uno di due punti distinti della retina. In questo modo l'unità diventa un rilevatore di quella feature, dotato di una certa invarianza rispetto la posizione della feature sulla retina (Figura 16). Questo esempio vuole solo dare un idea generale delle possibilità che si hanno costruendo modelli neurali. Ovviamente con una rete più complessa si potrebbe avere un sistema che riconosce più feature per più di due possibili posizioni di esse.
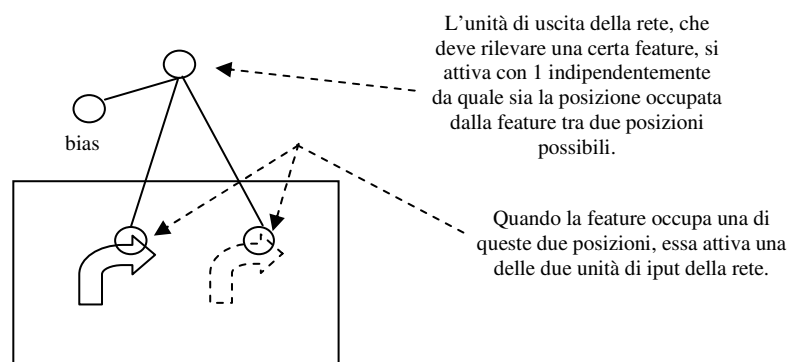
Figura 16 Il perceptrone che implementa la rete OR può essere usato come modello di una componente di un sistema percettivo.

Il perceptrone che implementa la funzione AND, invece, può essere utilizzato come modello di un sistema di riconoscimento di target. Sistemi di riconoscimento di questo genere sono ad esempio ipotizzati negli esperimenti di "matching to target" di psicologia dove i soggetti debbono identificare un particolare item, visto in precedenza, tra una serie di item distrattori. Il perceptrone che implementa la funzione AND può essere interpretato come segue. La prima unità di input si attiva solo se l'item da riconoscere corrisponde ad un certo template (modello). La seconda unità di input si attiva solo se viene foveato un item simile al template. A questo punto il perceptrone

fungerà come sistema di riconoscimento in quanto esso darà 1 in uscita solo se l'item foveato è uguale all'item da cercare (Figura 17).
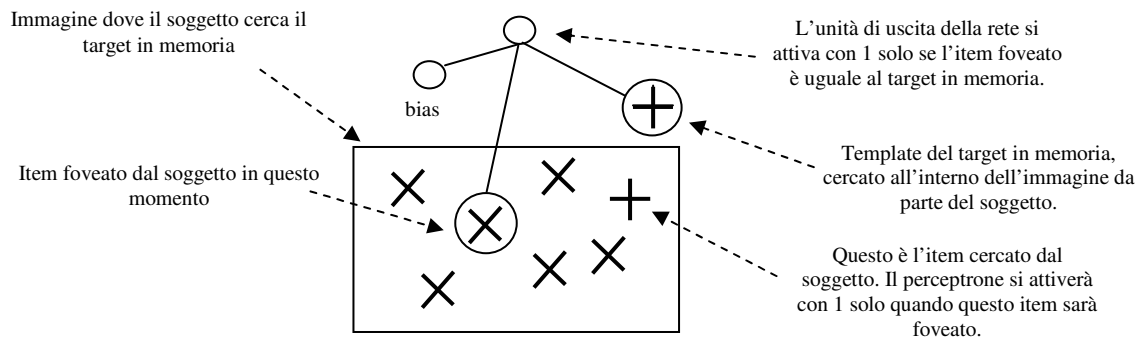
Figura 17 Il perceptrone che implementa la rete AND può essere usato come modello di un sistema "matching to target".

Purtroppo la brevità di questa introduzione alle reti neurali impedisce di presentare un modello neurale più sofisticato. Comunque già questi semplici modelli presentati fin qui danno un'idea di alcuni vantaggi presentati dai modelli computazionali rispetto ai modelli "verbali"normalmente utilizzati dalla psicologia:

- Il modello deve essere specificato fin nei minimi dettagli per funzionare. Questo obbliga a fornire *"teorie" complete*, ed obbliga ad interrogarsi in profondità sui termini ed i concetti che si usano normalmente per capire i fenomeni psicologici osservati. I modelli impongono quindi un rigore teorico non imposto dai modelli qualitativi.
- Il modello può essere "*fatto girare*", cioè possono essere fatti degli esperimenti con esso in modo da verificare se il *modello sia coerente*.
- Il modello può essere fatto girare producendo delle *predizioni* che poi posso diventare oggetto di studio sperimentali.